Reproducing Nutriscore: A Machine Learning approach

Sebastian Campbell & Giridhar Ande

March 18, 2023

Reading this document

You may read this report in the format of your choosing:

HTML | PDF | GitHub

Task description

We'd like to see if we can predict Nutriscore from foods using the OpenFoodFacts API. With a suite of clustering and classification models, we'd like to work backwards to see if we can work out what determines Nutriscore. While this exercise seems pointless in itself, the same technique could be used to work backwards to calculate closed source indices, such as those used by financial industries.

We'll try a naïve approach to start with, using nutritional factors to predict Nutriscore using an array of unsupervised clustering techniques to see if they form natural groups. [1]

[1] If these groups exists, I suspect this is how the index was created in the first place.

Afterwards, we'll use supervised classification techniques to create Nutriscore models. We'll cross-validate them and check them against a validation set.

Data description

Our initial plan was to use python `openfoodfacts` library to pull in data directly from the API. Unfortunately, the API was unreliable and had frequent down periods. We opted to use a predownloaded dump of the OpenFoodFacts data from Kaggle. The kaggle data comes in the form of a zipped tab-separated file (.tsv). When extracted, it's about 1GB in size and contains 356 027 rows and 163 columns.

Processing

In order to start using this data, we first had to filter and process it.

Nutriscore is one dimension available on OpenFoodFacts among others, including nutritional data per 100g [2]:

- Energy (kJ)
- Protein
- Sugar
- Sodium
- Salt
- Saturated fat
- Fat
- Carbohydrates

[2] We chose these nutrients as they're the most common ones to find in nutritional information. Stepping out of these moves you from 90% of foods having them, to 90% without.

It also contains tags for all foods, making it easy to specify food types. Calculating the Nutriscore values for everything would be tedious and take a long time to analyse so we're going to look at a few products:

- Breakfast cereals
- Iced teas[3]
- Biscuits and cakes

[3] Iced tea varies a lot in sugar content as it's often geared to the health market and so has options with either low sugar, or where some of the sugar has been replaced with artificial sweeteners.

All of these foods are highly variable in sugar and fat content so it should give us an idea of what causes Nutriscore to change.

Filtering

Even after reducing our dataset, we still noticed some unusual values. In particular, we noticed some salt contents that were peculiarly high.[4] We chose to remove all values with higher than 2% salt and the corresponding value for sodium.[5]

Given that the data we had retained was relatively complete, we opted to discard any rows with missing values rather than imputing. This caused a loss of ~15% of our data.

[4] Hello Panda may not be the healthiest treat, it's unlikely to contain more than its own weight in salt.

[5] Table salt is ~40% sodium, so our threshold was $(2 \times 0.4 = 0.8)$

Exploration

Nutrients

The first thing to look at when you have a new dataset is distribution. Let's look at overall distributions of each of our dependent variables.

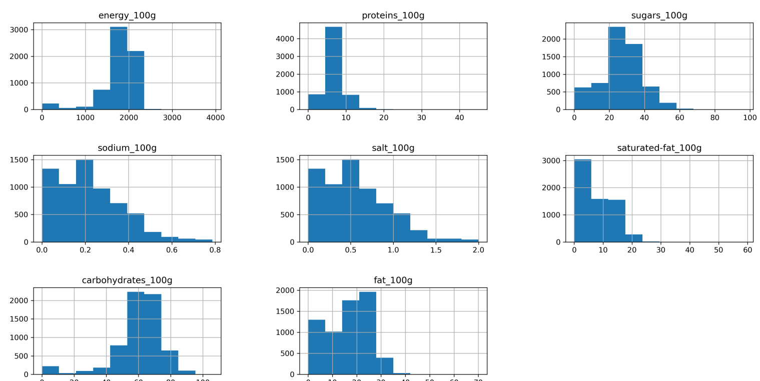


Figure 1: Nutrient distributions of biscuits, iced tea and breakfast cereals

A number of these have large outliers, but after sodium and salt were taken out in the processing step, none of them seem too terrible.[6]

The next stage of describing out nutrients would be to split these by type of food.

[6] Turns out the reason there's a huge outlier here is the presence of some dried coconut that counts as a biscuit. We didn't remove it as there are a number of biscuits from Brittany not too far away with astonishing levels of saturated fats.

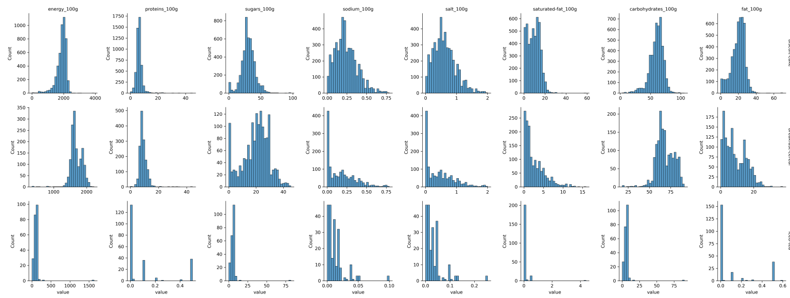


Figure 2: Nutrient distributions of a subset of food categories

There are some very clear patterns in the nutrient distributions of food category.[7] Iced tea behaves quite differently to the others as it has virtually no nutrients other than sugar and energy. Biscuits seem to have the most normally distributed nutrients, while cereals seem to have a number of healthy ones with low sodium and fat but a large spread with more of those nutrients.

[7] Please note that the x-axes are free. I wasn't able to fix it per column and it would have been pointless to fit it for the whole grid.

Nutriscores

Now that we've looked at the nutrients, we'll turn our attention to the nutriscores.

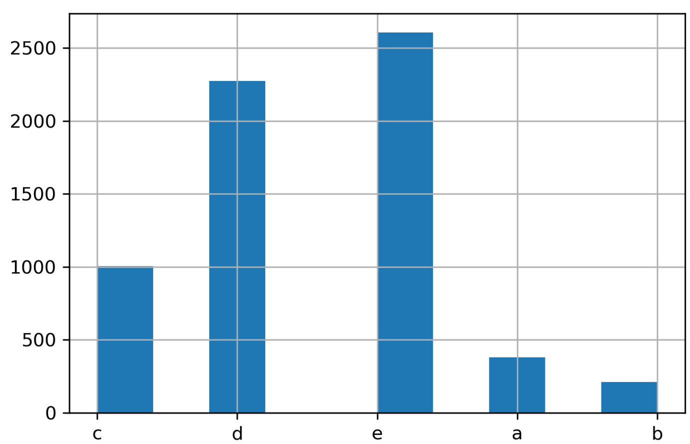


Figure 3: Nutriscore distribution in our data

Out data is mostly unhealthy food, it would seem. We have a lot of foods with scores of D and E, but not so many with scores of A and B. Given that most of the data in OpenFoodFacts is processed food, this isn't surprising - but our choices of category are certainly not helping our cause.

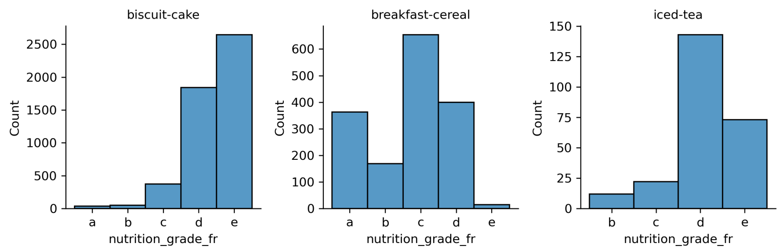


Figure 4: Nutriscore distribution by food

Splitting up the Nutriscores by category, we can see that iced teas are generally unhealthy, with not a single one getting a score of A. Biscuits and cakes show an expected distribution as do breakfast cereals as there's a lacuna between All-Bran and Coco Pops.[8]

[8] I suspect that the reason that there are so many Cs are unhealthy cereals adding fibre to their cereals to get to the first "healthy-ish" tier.

Unsupervised clustering

The first question we wanted to answer was "Do Nutriscores form natural groups?". While there is obviously an underlying model linking foods to Nutriscores, the question could be otherwise formulated: "If a machine creates clusters without knowledge of Nutriscores, will the clusters resemble Nutriscores?"

Method

We chose 10 unsupervised clustering models:

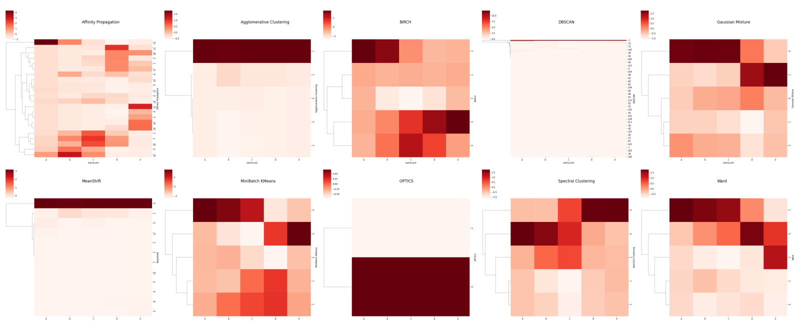
- MiniBatch KMeans
- Affinity propagation
- MeanShift
- Spectral clustering
- Agglomerative clustering (Ward linkage)

- Agglomerative clustering (average linkage)
- DBSCAN
- OPTICS
- BIRCH
- Gaussian mixture

Where possible, we limited them to 5 clusters. We then ran the models on our training data and compared them to our nutriscores. As there's no actual mapping between which cluster corresponds to which score, we've created heatmaps normalising the correspondances[9] by nutriscore and with rows rearranged using hierarchical clustering.

[9] If you dont' normalise the correspondances, you'll end up with really low values for clusters with few members, even if the correspondances are good.

Results



There's a lot to break down here. The models that performed poorly:

- Agglomerative clustering (average linkage)
- DBSCAN
- MeanShift
- OPTICS

These models tended to push everything into one cluster. It might be possible to fix this by adjusting the parameters.

The other models seemed to able to cleanly split scores A and B from D and E. Affinity propagation is worth further investigation as while it has multiple clusters, it seems to have a cluster which identifies each Nutriscore

Conclusion

Given that these we found in an unsupervised manner, it's safe to say that Nutriscores are natural groupings found in the data.

Classification

We looked at 6 classification models:

- K nearest neighbours
- Random forest
- Decision tree
- Linear SVM[10]
- SVM with RBF
- Ridge regression

[10] We had to reduce the regularisation parameter (C) to 0.01 for both SVM models as the default value of 1 was taking hours to fit.

Each of these models was run with default parameters in `sklearn` and cross-validated using a 5-fold method. 20% of the dataset was randomly selected as a validation set in addition to the cross-validation. All

Cross-validation results

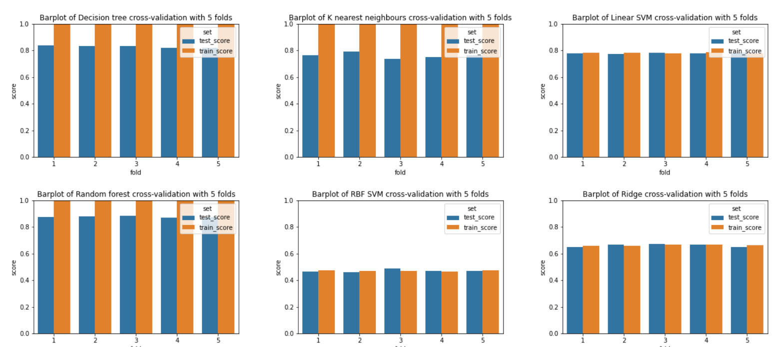


Figure 5: Cross-validation of all classification models using 5 folds

The SVM models performed particularly poorly - both in their training and during the cross-validation. This is almost certainly because we had to cripple them with low regularisation parameters or else they simply took too long to run.

Ridge regression is notable for having very similar values for training and cross-validation. There's no overfitting, but its overall performance with its current parameters is too low to consider.

K nearest neighbours, decision trees and random forests had near perfect performance on their training sets and good performance in cross validation. The winner is clearly *random forest*, having the highest cross-validation scores.

Validation

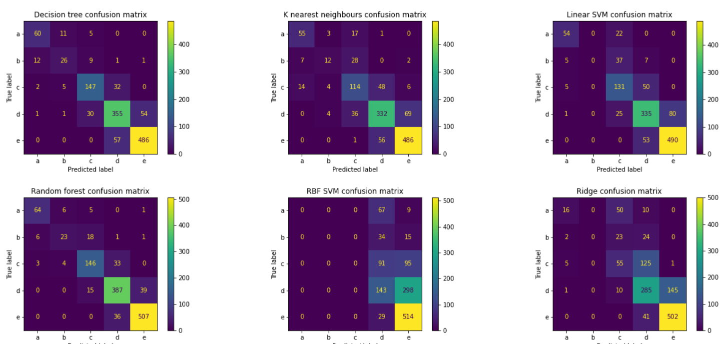


Figure 6: Unnormalised confusion matrices for each of the models predicting Nutriscore from nutritional information

Conclusions

Future improvements[11]

- Find more foods with Nutriscore A and B
- Perform Feature importance to to more easily find which variables are more important
- Normalising variables may have led to better results for sodium/salt
- Searching variable space to optimise parameters

[11] We *could* have added more things - but aren't you tired of reading it already?